Remote logs for ONT in GPON networks

Afonso Augusto Romão V. Alvim
afonso.alvim@hotmail.com

Fernanda Yumi Matsuda
Aline Cristine Fadel
André Berti Sassi
CPqD Foundation
Rod Campinas - Mogi-Mirim (SP-340) - km 118,5
13086-902 - Campinas, SP - Brazil
{ alvim, fmatsuda, afadel, asassi }@cpqd.com.br

Marcos Perez Mokarzel
Instituto Nacional de Telecomunicações - Inatel
P.O. Box 05 - 37540-000
Santa Rita do Sapucaí - MG - Brazil
mpmoka@inatel.br

***Abstract*— In this paper, we propose a new scheme to remotely recover execution information from ONT in GPON, using two new managed entities to complement OMCI MIB with remote log features. The first entity provides means to ONT to send log information to OLT. The second provides means to OLT to adjust ONT date and time, which is employed in log synchronization. We've implemented this log scheme in real equipment proving that cost-benefit ratio is positive and remote logging can be employed both in development and operation time.**

***Index Terms*— Remote log, GPON, ONT, ITU-T G.984.**

I. INTRODUCTION

Usually, logs are the only way for developers to discover problems and abnormal events in an embedded system. They are the register of all information regarding to software procedures and hardware feedbacks. This information is saved as tickets in a file that can be analyzed by testers and developers.

Advantages of log systems are discussed by Karpov [1]. It is shown that in systems with multiple terminals, such as networks, logs have a particular importance since, in this equipment, multiuser and multithread algorithms are normally employed. In such applications, errors often occur when a lot of threads are created or when there are synchronization problems. Errors in parallel programs are rather difficult to find. A good method of detecting such errors is periodical logging of systems which relate to the error and examining of the log's data after a program crash [1].

Watterson [2] investigated the feasibility of using a logger system in execution time for embedded software to assure that the system is in compliance with its requirements. Issues regarding monitoring are studied with experimental tests involving specification and implementation of test platforms. Based on tests' results, Watterson proposes some requirements for an efficient logger system: scalability, modularity, usability and low execution interference.

As proposed by Fryer in [3], monitoring systems should be Real-Time and Non-Intrusive (RTNI), which means that "the instrumentation mechanisms must be transparent to behavior of software". This goal hasn't been achieved yet, but, as we prove in our paper, we have got good results with precision lower than 1 second and increasing, in the worst case, processing time in 15%.

An important consideration was made by Watterson and Heffernan in [4]. They confirm Fryer's approach and also state that all monitoring system should be enough to observe all software behaviors. In our system, we provide a way to analyze ONT's and OLT's behavior simultaneously, providing a very efficient debugging tool.

GPON (Gigabit-capable Passive Optical Networks) system, standardized by ITUT G.984 [5]-[8], is composed by one OLT (optical line terminal) placed in the central office and several ONTs (optical network terminals) at the users' premises.

Nowadays, ONT maintenance usually involves a field engineer going to user's premises to execute tests and get data. Most of the time, it is impossible for engineers to detect the problem remotely, without any log. It is worse if the issue is in software, because all data collected by the field engineers may not help to find and fix the issue.

Many systems provide local log information, partially solving the problem, but still requiring physical access to the equipment. So, a system where ONT can remotely generate and send logs to the OLT might be helpful, facilitating detected errors analysis at the central office, reducing the costs of sending a field engineer to the user's premises. In addition, if it is a software issue, it will be simple to send the log to the software engineer to locate and correct the issue.

GPON standard ITU-T G.984.4 [8] describes the ONT management and control interface (OMCI). The OMCI is a management information base (MIB) and a set of tools used to keep ONT and OLT MIBs synchronized. In the OMCI MIB, there are many entities for operation, administration and maintenance (OAM), but no entity provides logging feature.

In Section II, it is described the new managed entities (MEs): *ONT Logger* and *Date and Time* and, in Section III, our implementation in a real GPON system. In order to prove that this scheme doesn't have any negative influence over the whole system, CPU time was measured and some results are shown in Section IV. The analysis results are presented in

Section V, and finally, in Section VI, we present our conclusions and propose new researches for future works.

II. ONT Logger and Date and Time entities

Following [8], *ONT Logger* and *Date and Time* are MEs that can be synchronized using OMCI process. From the viewpoint of ONT configuration, these entities are connected, in a one by one relationship, with ONT-G entity and are part of equipment MEs.

A. ONT Logger entity

This ME represents the ONT's logger as a slave ticket provider.

An instance of this ME is automatically created by the ONT after initialization. After the creation of this ME, associated attributes are updated according to the data within the *ONT Logger* buffer capacities.

Relationships

This managed entity is related directly to the ONT-G entity.

Attributes

Managed Entity id: This attribute provides a unique number for each instance of this managed entity. There is only one instance and it has the number 0x0000. (R) (mandatory) (2 bytes)

Is Logger Active: This attribute is used to identify if the logger is active or inactive in ONT:
0: Logger is inactive;
1: Logger is active;
(R, W) (mandatory) (1 byte)

Ticket Mask: This attribute is a mask of bits that identifies which logger types must be saved in the log by ONT. Table I describes the bit mask that is used individually or combined. Bits set will be logged. (R, W) (mandatory) (2 bytes)

TABLE I
Ticket bit mask.

Bit	Log type	Description
1	ERROR	Fatal error, impossible to be fixed
2	WARNING	Minor error, fixed by the system
3	NULL	Access NULL address
4	START	Function starts
5	END	Function ends
6	STATE	Finite state machine states
7	SUCCESS	Operation success
8	INFO	Ordinary information
9	MANAGER	Manager action
10	HARDWARE	Hardware action
11	ALARM	System alarm sent to operator
12	MEMORY	Memory allocation and deallocation
13	OPERATOR	Operator command
14	COMM	OLT communication
15	EVENT	Event (Trap) to operator
16	RESERVED	

Log Buffer: This attribute is a ticket buffer, it informs buffer size in the Get message and buffer contents in subsequent Get Next messages. (R) (mandatory) (M bytes, where M is the ticket size)

Actions

Get: Get one or more attributes.
Set: Set one or more attributes.
Get Next: Get next ticket in the buffer.

Notifications

Attribute value change: This notification is used to report autonomous changes to the attributes of this managed entity. The attribute value change (AVC) notification should identify the attribute changed and its new value. The list of AVCs for this managed entity is given in Table II.

TABLE II
AVC list for ONTLogger

Number	Attribute value change	Description
1…2	Reserved	
3	Log Buffer	This AVC indicates that a set of tickets is available in the ONT buffer
4…16	Reserved	

B. Date and Time entity

This managed entity represents the ONT real time clock.

An instance of this managed entity is automatically created by the ONT after initialization. After the creation of this managed entity, the associated attributes are updated according to the OLT real time clock information.

Relationships

This managed entity is related directly to the ONT-G entity.

Attributes

Managed Entity id: This attribute provides a unique number for each instance of this managed entity. There is only one instance and it has the number 0x0000. (R) (mandatory) (2 bytes)

Year: This attribute contains the year in current date (R, W) (mandatory) (2 byte)

Month: This attribute contains the month in current date (R, W) (mandatory) (1 byte)

Day: This attribute contains the day in current date (R, W) (mandatory) (1 byte)

Hour: This attribute contains the hour in current time (R, W) (mandatory) (1 byte)

Minute: This attribute contains the minute in current time (R, W) (mandatory) (1 byte)

Second: This attribute contains the second in current time (R, W) (mandatory) (1 byte)

Uptime: This attribute contains the time in milliseconds since the last ONT reset (R) (optional) (4 bytes)

Actions

Get: Get one or more attributes.
Set: Set one or more attributes.

Notifications

None.

III. GPON Implementation

In order to test and prove that this remote log doesn't have high impact over equipment performance, we've implemented it in CPqD's real GPON system.

This system conforms to ITU-T G.984 recommendations [5]-[8], including OMCI with its MIB, extended with the MEs *ONT Logger* and *Date and Time*, described above.

These two entities are created automatically during ONT initialization. In the MIB synchronization process, OLT must set ONT's date and time with its own current time, allowing the system to synchronize logs generated by OLT with logs from ONT.

When *ONT Logger* is created, two buffers are allocated. *Buffer A*, is used to store new tickets and *Buffer B*, to transmit stored tickets to OLT. These buffers are FIFO (First in, First out) lists, limited in size and time. Size and time limits are a project decision. The size limit depends on ONT memory size, and in our implementation, we've set the size to 10 tickets. The time limit is only used if ONT is generating few tickets. In this case, tickets may wait for a long time in the buffer and may lose objectivity. Once the time is stamped in the ticket when it is created, buffer time limit has no influence over time precision. In our system, we use 5 seconds as the time limit.

A. Buffer utilization

After ONT's MIB is reset, buffers are empty and the ONT starts to create log tickets and to store them in the *Buffer A*. When the buffer size or time limit is reached, ONT freezes this buffer and generates an AVC for Log Buffer attribute in the *ONT Logger* ME. After that, ONT starts writing new tickets in the *Buffer B*.

OLT reads *Buffer A* using *Get* and *Get Next* actions. When *Buffer B* size or time limit is reached, the buffer is frozen, another AVC is sent to OLT and ONT starts writing new tickets in *Buffer A*, cyclically.

It is up to the OLT to read the buffer after AVC notification. As the OLT may take a long time before reading the ONT's buffer, the other buffer may also be entirely filled. In this case, the ONT erases the first buffer, creates a new WARNING ticket reporting that tickets were lost in the beginning of the buffer and continues writing new tickets.

IV. Tests and Results

A. Test infrastructure

The scheme proposed in this paper was implemented and tested in a real GPON system. The OLT manages up to 1024 ONTs and supports OAM through multiple Command Line Interface (CLI) terminals and Simple Network Management Protocol (SNMP) via Ethernet port or RS-232 serial port.

The OLT has a Logger component that formats the log tickets and sends them to a remote PC through UDP/IP packets. The ONT has the same Logger component, but the logs are forwarded to the OLT by OMCI, using the *ONT Logger* ME described above. Both OLT and ONT software were written in C language.

In the remote PC, WinLogger is the software responsible for handling the received tickets. It displays reports and saves the tickets in files. The reports are saved in tables that can be opened in many different spreadsheets, allowing better visualization of statistical data.

B. Test methodology

For the tests, the configuration of our GPON system was made using scripts in OLT's CLI. Once each GPON operation has its own needs (for instance, some operations need more CPU time, while others need more network traffic) we've defined a script including many different operations, with different needs. The script follows the sequence below.

1. Activate Link 1.
2. Register ONT 1 in Link 1.
3. Activate ONT 1.
4. Set *Date and Time* in ONT 1.
5. Activate *ONT Logger* in ONT 1.
6. Create 10 Ethernet flows in ONT 1.
7. Activate all Ethernet flows in ONT 1.
8. Deactivate ONT 1.
9. Deactivate Link 1.

The step 5, which is an OMCI Set command to the *ONT Logger* ME to activate its functionality, was skipped in half of the test executions. This way, it is possible to evaluate the ONT Logger impact in OLT's performance.

The methods used to extract performance data and to generate the results below are the same used in [9]. Our OLT software also uses the same scheduling mechanism. Moreover, the tasks are assigned to two priority queues, depending on its type: the hardware callbacks, command execution and OMCI response handling tasks are assigned to the low priority queue and a main internal process, which runs on a periodic basis, is assigned to the high priority queue.

Only the results obtained in step 7 are considered here, because our ONT doesn't generate a significant amount of logs in other steps. The activation of Ethernet flows is the worst case of ONT Logger in our GPON system.

The test was executed four times for each case: with and without Logger.

C. Test results

The Fig. 1 shows the average of waiting time for tasks assigned to the high priority queue in both cases.

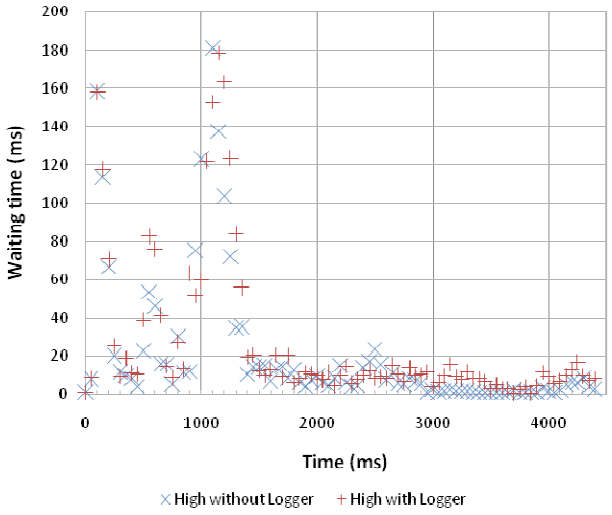

Fig. 1. Average waiting time for tasks assigned to high priority queue.

The Fig. 2 shows the average length of high priority queue during the flow activation.

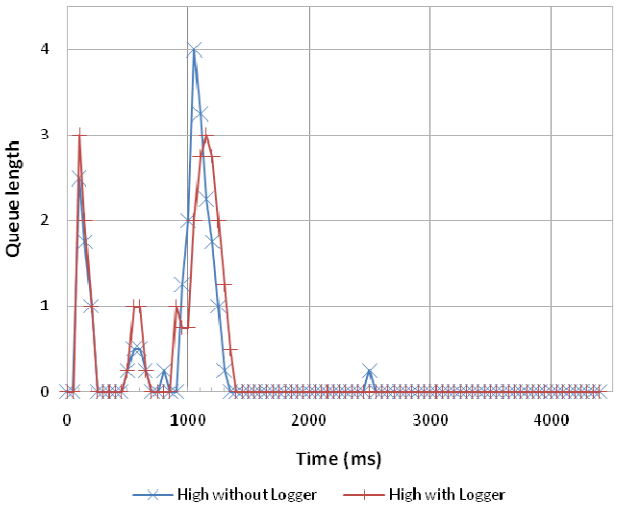

Fig. 2. Average high priority queue length.

The Fig. 3 and the Fig. 4 show the corresponding results for low priority queue, i.e., waiting time and queue length, respectively.

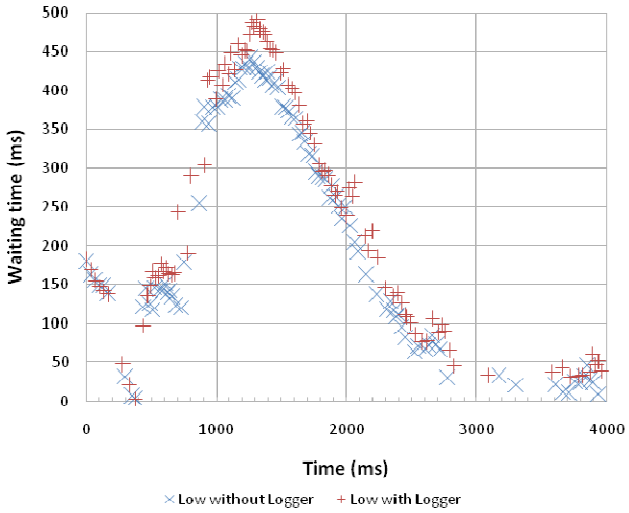

Fig. 3. Average waiting time for tasks assigned to low priority queue.

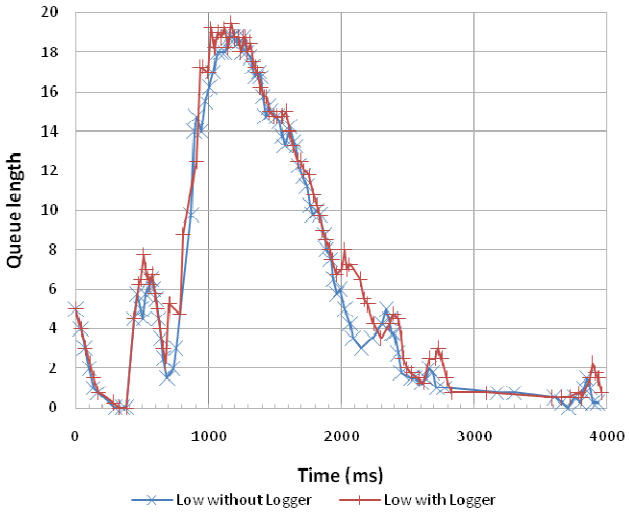

Fig. 4. Average low priority queue length.

The Table 1 shows the comparison between OLT's software performance when ONT Logger is active and when it's not. It

TABLE I
COMPARISON OF PERFORMANCE PARAMETERS WITH AND WITHOUT LOGGER.

Results	***ONT Logger* state**		**Relative increase**
	Active	**Inactive**	
Average waiting time in high priority queue (ms)	26.6	22.3	19%
Average waiting time in low priority queue (ms)	241.9	220.3	10%
Average queue length in high priority queue	0.28	0.26	10%
Average queue length in low priority queue	8.3	7.9	6%
Average response time (ms)	5910	5894	0.28%
Standard deviation of response time (ms)	28.7	28.8	0.67%

also shows the relative disadvantage of using the ONT Logger for each parameter.

V. Results evaluation

Comparing the graphs in Fig. 1 and Fig. 2, which represent the behavior of high priority queue, the average difference of waiting time and queue length between each case is small. In absolute values, Table 1 shows the Logger increases only 4.3ms in waiting time for the high priority queue and the queue has only 0.02 messages more, in average.

However, the graphs in Fig. 3 and Fig. 4, which represent the behavior of low priority queue, show a slight increase in waiting time and queue length in some specific and short time interval. These time intervals correspond to ONT Logger synchronization in OLT and happen around 500ms, 900ms, 2000ms and 2700ms.

The average waiting time of low priority queue when the ONT Logger is active has increased 10% in relation to the tests without ONT Logger activation and the queue length has increased 6%.

On the other hand, the response time has not significantly changed. Its average increased only 16ms, which is considerably lower than standard deviation of our results.

VI. Conclusion

In this paper, we present an alternative approach to recover log information from a remote ONT in a GPON system. We use OMCI infrastructure and, creating two new entities, we were able to get ONT log tickets in OLT, avoiding the need of physical access to ONT. In OLT, the ONT tickets joins OLT tickets and are send to a remote computer through UDP/IP.

Date and Time ME provides means to OLT synchronize ONT time with its own time, which is important to join tickets generated by both equipment in a coherent sequence. Tickets were sent from OLT to ONT using *ONT Logger* ME.

In real GPON equipment, we've made tests and proved that cost-benefit ratio was positive. After ONU log activation, average waiting time in OLT scheduler queue increased only 21.6ms in the worst case, average queue length increased at most 10% and the response time didn't increased at all. In our system, these costs were acceptable and impact on the performance was not felt by the operator or by the subscribers.

References

[1] A. Karpov (2008, Nov.), "Building of systems of automatic C/C++ code logging," published by Intel Corp. Available: http://software.intel.com/en-us/articles/building-of-systems-of-automatic-cc-code-logging/.

[2] C. Watterson, "A Monitoring Approach to Facilitate Run-time Verification of Software in Deeply Embedded Systems" PhD thesis, University of Limerick, 2010.

[3] R. Freyer, "FPGA Based CPU Instrumentation for Hard Real-Time Embedded System Testing". SIGBED Rev., vol. 2, no. 2, pp. 39–42, April 2005.

[4] C. Watterson and D. Heffernan, "A runtime verification monitoring approach for embedded industrial controllers" in IEEE Int. Symp. on Industrial Electronics (ISIE), Cambridge, 2008, pp. 2016–2021.

[5] Gigabit-capable Passive Optical Networks (GPON): General Characteristics, ITU-T Rec. G.984.1, 2003.

[6] Gigabit-capable Passive Optical Networks (GPON): Physical Media Dependent (PMD) layer specification, ITU-T Rec. G.984.2, 2003.

[7] Gigabit-capable Passive Optical Networks (GPON): Transmission convergence layer specification, ITU-T Rec. G.984.3, 2008.

[8] Gigabit-capable Passive Optical Networks (GPON): ONT management and control interface specification, ITU-T Rec. G.984.4, 2004.

[9] A. Alvim et al. "Time survey on embedded software using remote IP log system: a GPON study case," in Int. Workshop on Telecommunications, Rio de Janeiro, Brazil, 2011.